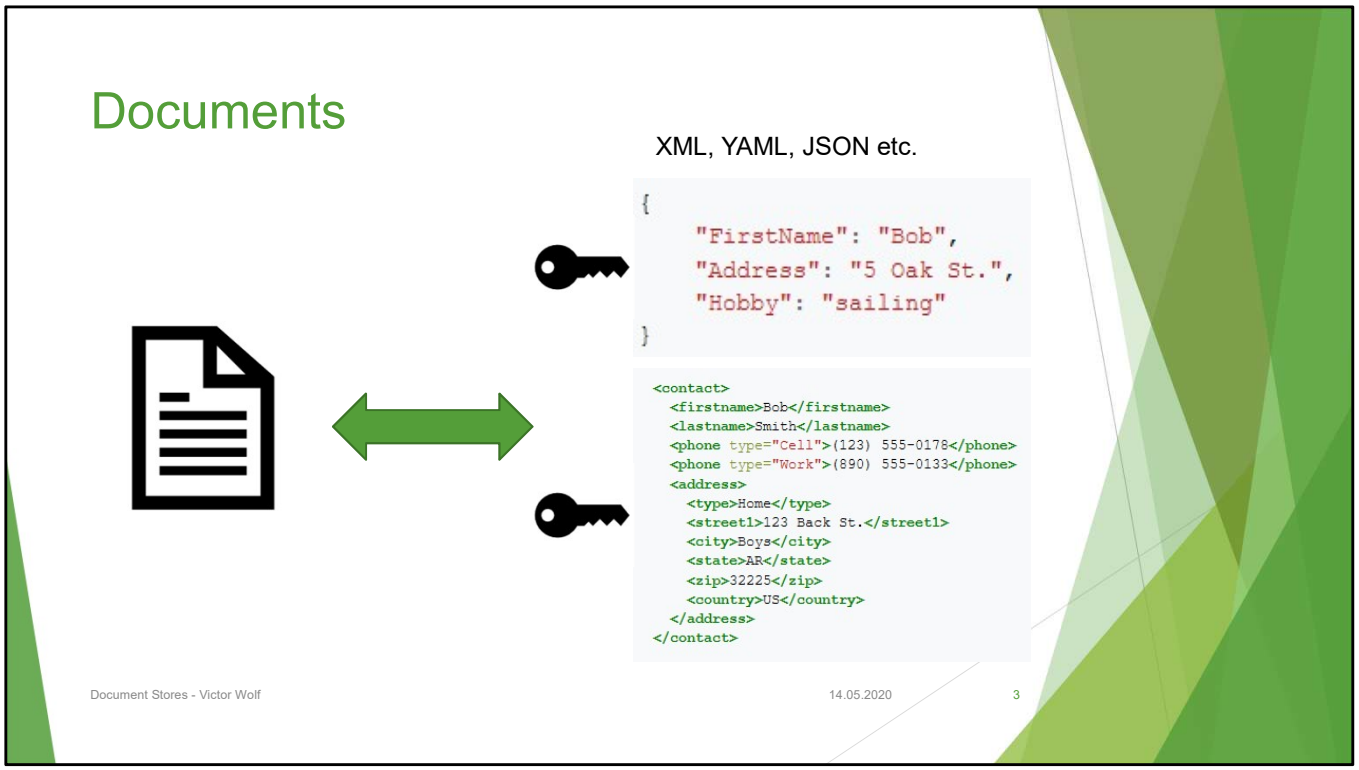
Document Stores

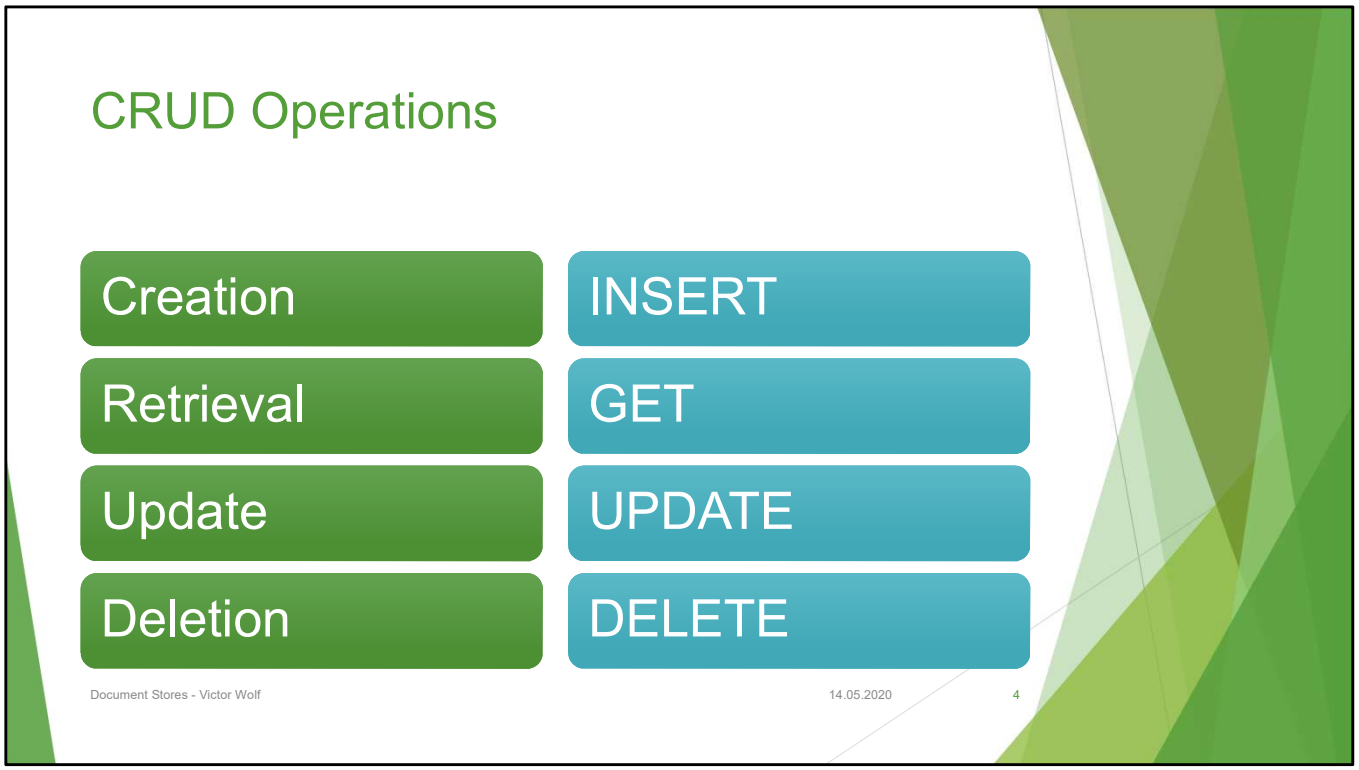
Mit Fokus auf MongoDB

Gliederung

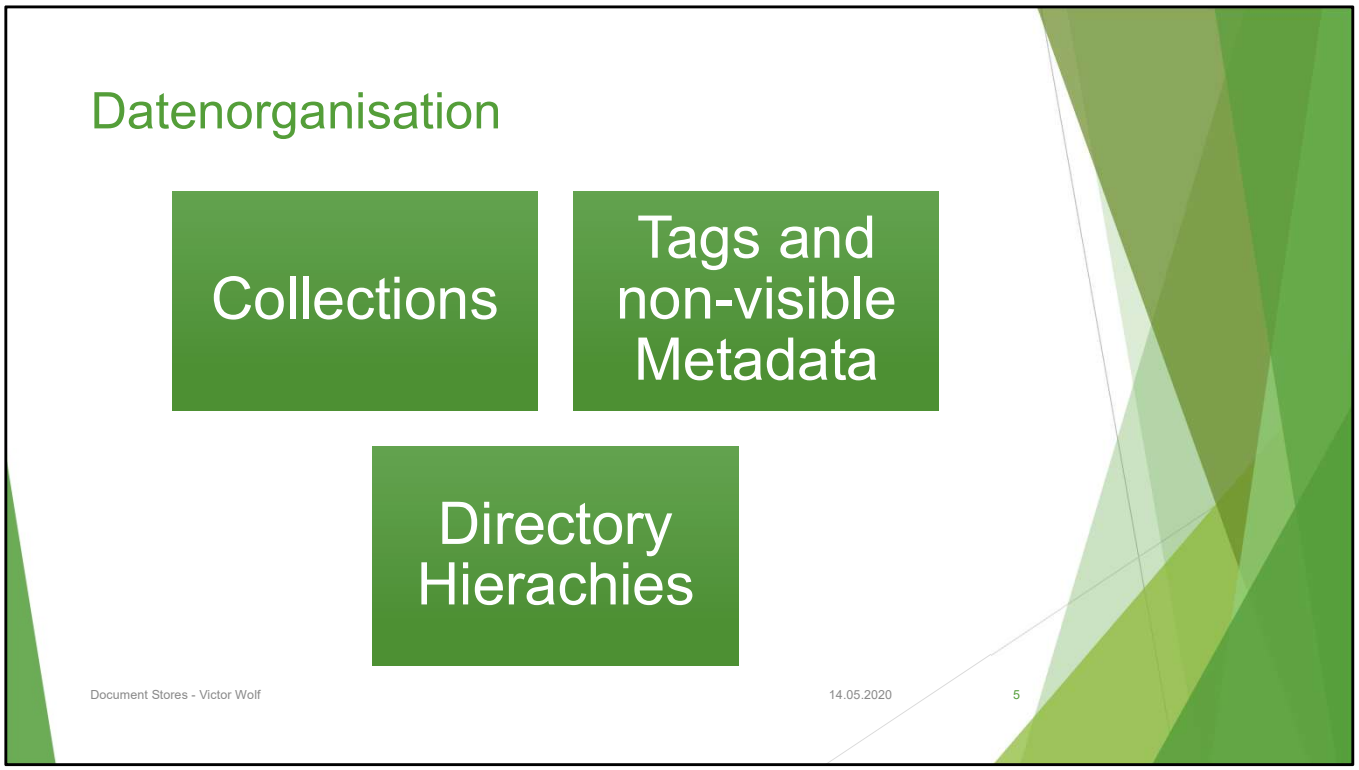
1. Allgemein
2. MongoDB
3. Live Demo
4. Fragen
5. Key Takeaways



1. The central concept of a document-oriented database is the notion of a *document*. While each document-oriented database **implementation differs** on the details of this definition, in general, they all assume **documents encapsulate and encode data (or information) in some standard format** or encoding. Encodings in use include XML, YAML, JSON, as well as binary forms like BSON.
2. Nähe zum Internet
3. Ähnlichkeit zu einem **Objekt** aus der Objekt orientierten Programmierung
4. Jedes Dokument bekommt eine eindeutige ID mit der es eindeutig identifiziert werden kann.
5. Extrembeispiel Document Store: key aus zahlen mit plain text aber das stimt nicht ganz
6. Können Metadaten enthalten
7. Sind somit semi strukturiert.
8. Spezialisierte Key-Value Datenbank. Es gibt Felder und Werte diese können sich in der Struktur aber unter den Dokumenten unterscheiden!
9. Durch die ähnlichkeit zu Objekten direkte Konversion zu Objekten in den meisten Programmiersprachen möglich sowie Polymorphismus



- Basis für persistente Datenspeicherung
- Kommt euch das bekannt vor? Tipp: Denkt an das Internet
- Hier sind die Aktionen gemeint die mit der Datenbank abgedeckt werden können -> Mindestanforderung



- Variiert wieder je nach Implementierung, da es kein hartes Datenmodell gibt.
- Collections: Ähnliche Dokumente werden in einer Collection gruppiert Müssen aber nicht den selben Aufbau haben. Ähnlich zu Tabellen aus Relationalen Datenbanken
- Tags and non-visible metadata: Das Datenbanksystem definiert metadaten um dokumente besser gruppieren zu können. Z.B Zeitpunkt der Dokumentserstellung, Datenquelle oder Felder, die in diesem Dokument vorhanden sind. Diese sind ebenfalls wichtig um Abfragen auf der Datenbank zu optimieren, da wir kein zugrunde liegendes Schema haben.
- Directory Hierachies: Baumstruktur der Dokumente, ähnlich zu dateisystemen.
- Können sowohl physische als auch logische Representationen sein

Replikation und Partitionierung

“The NoSQL movement is motivated by horizontal scalability to leverage clusters of commodity hardware and minimization of the impedance mismatch between the data model of the application and the database. Horizontal scalability is addressed through partitioning and replication”

-> Dokumente sind in sich konsistent

Quelle: Felix Gessert, Norbert Ritter: Scalable Data Management: NoSQL Data Stores in Research and Practice

Beispiele für Document Stores

CouchDB ist von der apache software foundation.
elasticSearch ist ein spezieller document store sondern eher eine Suchmaschine, die aber auf Dokumenten basiert.
Couchbase von Couchbase Inc

Quiz

Welche von den vier ist der meist benutzte Document Store?

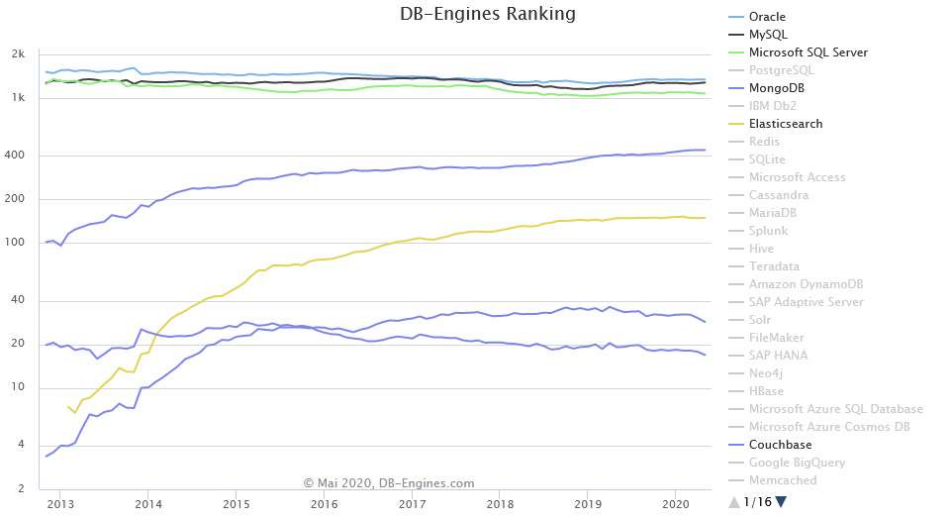
A: MongoDB

B: CouchDB

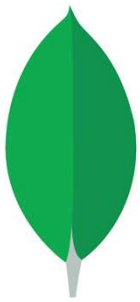
C: elasticSearch

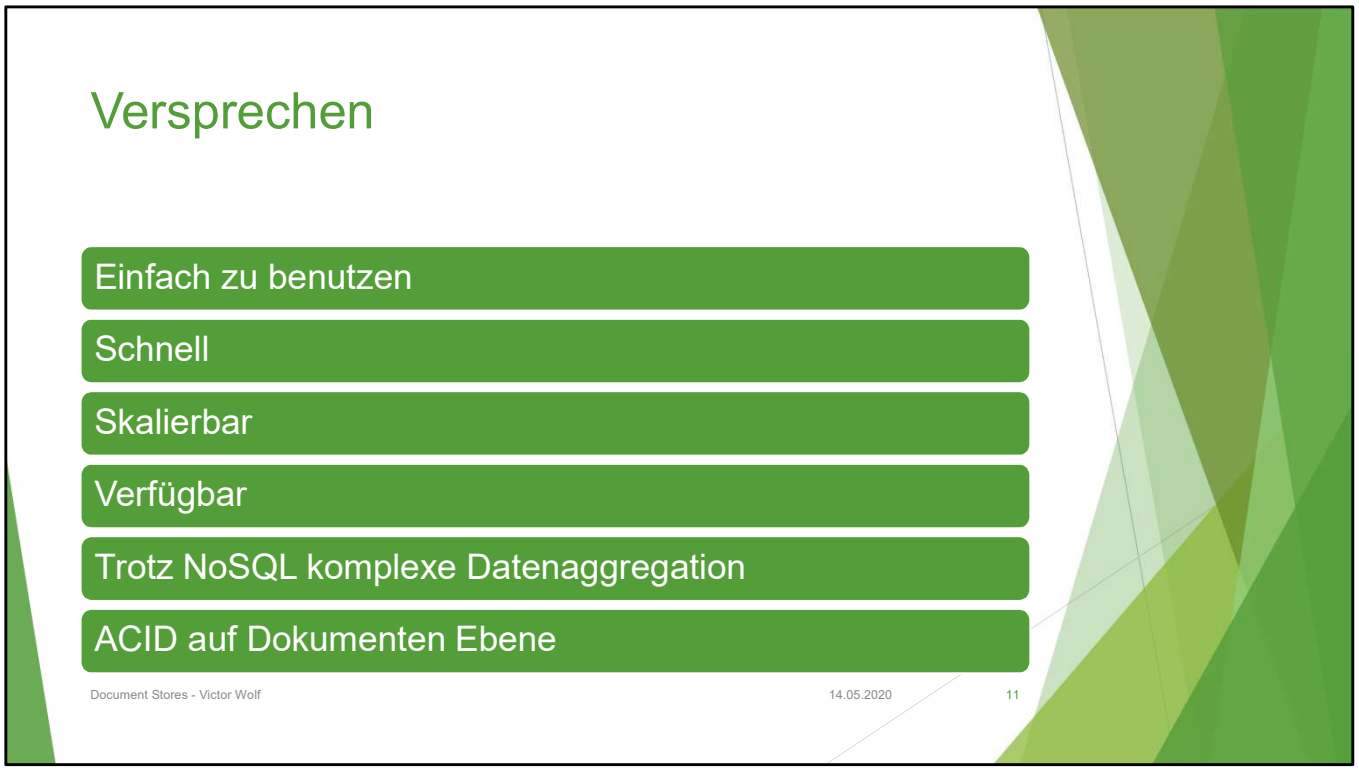
D: Couchbase

Quiz

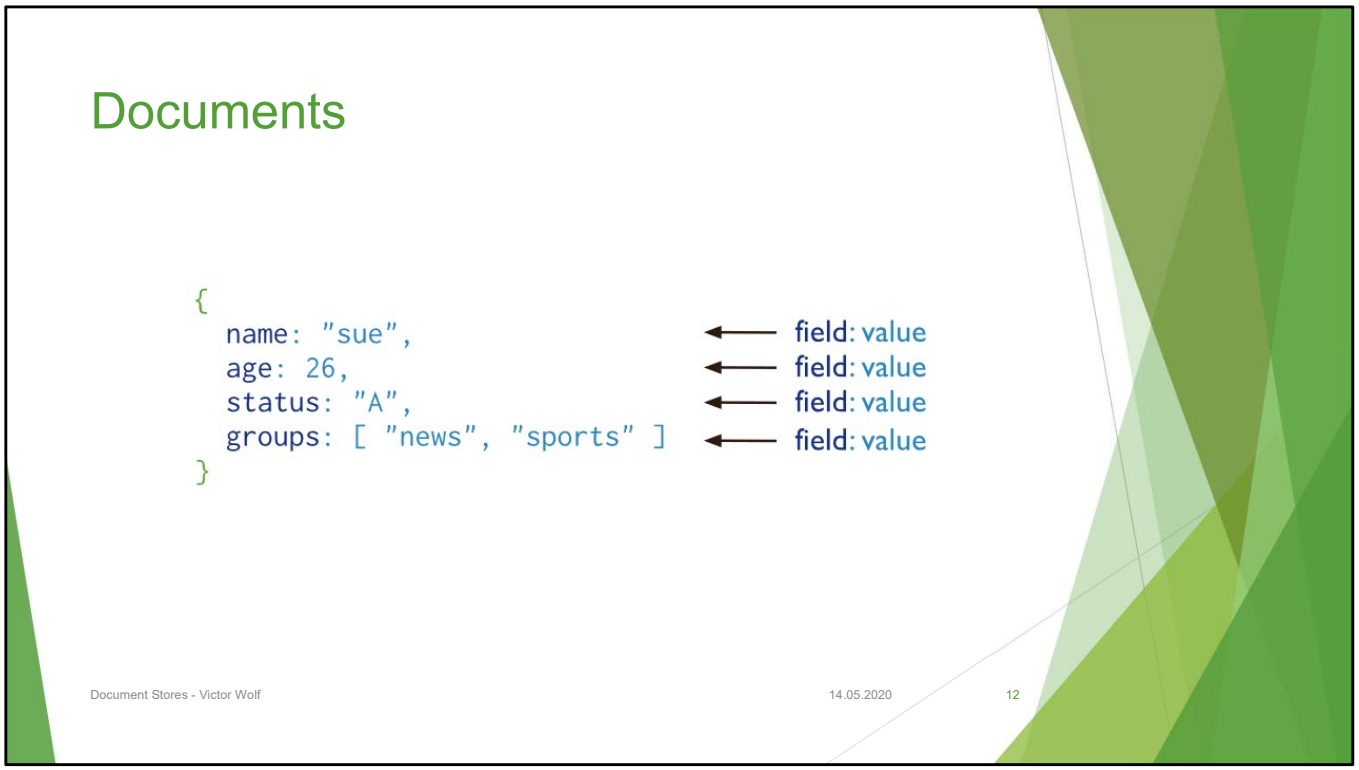

DB-Engines Ranking
2k
1k
400
200
100
40
20
10
4
2
2013
2014
2015
2016
2017
2018
2019
2020
© Mai 2020, DB-Engines.com
Oracle
MySQL
Microsoft SQL Server
PostgreSQL
MongoDB
IBM Db2
Elasticsearch
Redis
SQLite
Microsoft Access
Cassandra
MariaDB
Splunk
Hive
Teradata
Amazon DynamoDB
SAP Adaptive Server
Solr
FileMaker
SAP HANA
Neo4j
HBase
Microsoft Azure SQL Database
Microsoft Azure Cosmos DB
Couchbase
Google BigQuery
Memcached
1/16

1. Allgemein
2. **MongoDB**
3. Live Demo
4. Fragen
5. Key Takeaways





- ACID mitlerweile auch auf Operationsebene aber nur wenn man das will.



Dokumente in MongoDB folgen dem BSON Format, welches einfach ein Binarisiertes JSON Format ist.
PLUS das _id feld was immer hinzugefügt wird und als Primary Key dient.

API - Abfragesprache

```
mongodb@ubuntu:~/mongodb-linux-x86_64-2.6.0$ bin/mongo
MongoDB shell version: 2.6.0
connecting to: test
> db.adminCommand( { getLog: "global" } )
{
  "totalLinesWritten" : 34,
  "log" : [
    "2014-05-08T01:36:03.034-0400 [initandlisten] MongoDB ... ",
    "2014-05-08T01:36:03.036-0400 [initandlisten] db version v2.6.0",
    "2014-05-08T01:36:03.038-0400 [initandlisten] git version: ... ",
    ...
  ],
  "ok" : 1
}
>
```

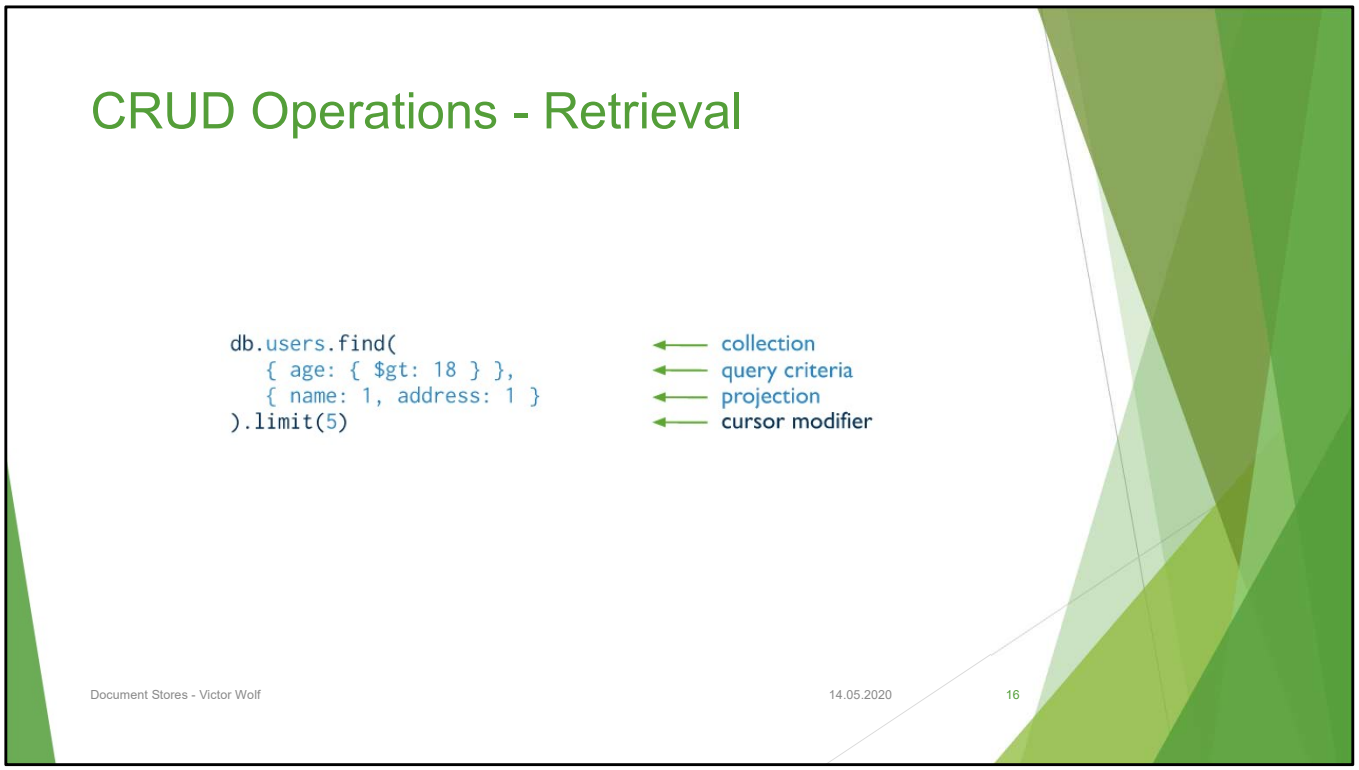

Websocket auf Port 27017 mit binärem Protokoll
MongoDBs integrierte Abfragesprache ist JavaScript
MongoDBs shell ist somit eine interaktive JavaScript Umgebung
Eigentliche Abfragen sind dann JSON

Treiber sind gut um die requests nicht selbst implementieren zu müssen.
Meiste aktuelle Programmiersprachen werden von MongoDB unterstützt.
Selbst Programmiersprachen wie Haskell werden von der Community unterstützt

CRUD Operations - Create

```
db.users.insertOne(          <-- collection
  {
    name: "sue",             <-- field: value
    age: 26,                 <-- field: value
    status: "pending"        <-- field: value
  }                                              } document
)
```


Ein Beispiel für einfügen.
Javascript ist wieder die Sprache.
Eigentliches Objekt was erzeugt werden soll ist in JSON.
Dokument Operationen sind atomic



Das Objekt, was wir übergeben ist diesmal ebenfalls JSON auch wenn es aussieht als würden wir dieses Objekt einfügen wollen, übergeben wir einfach eine Operation für die Datenbank in JSON.

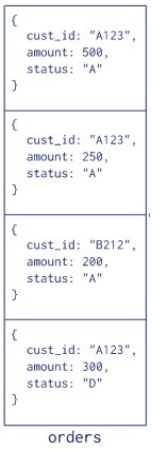
Jede Stage hat macht einen Schritt zur vollständigen Aggregation hin.
Die Befehle sind dabei JSON
Es gibt auch einen MapReduce Ansatz

Aufgabe

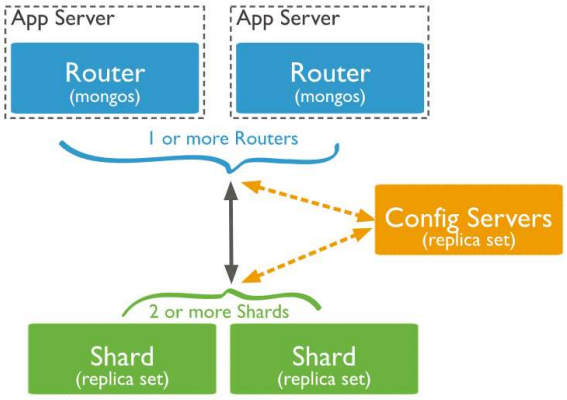
Ich möchte alle Dokumente deren amount Feld größer als 280 ist.

Tipp: Der „größer als“ Operator ist „gt“

```
db.orders.aggregate([
      { ---- : { ----- : { ---- : ---- } }}
])
```



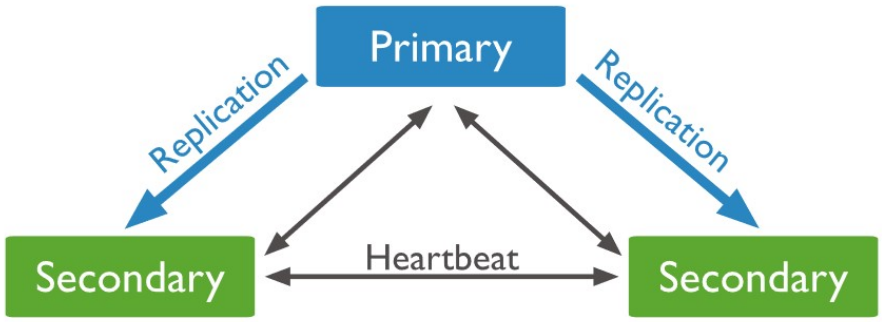
Sharding



Shard: Datenteil der aufgeteilten Gesamtdaten. Diese können nochmal repliziert sein. Daten werden pro Collection aufgeteilt. Jeder Shard besitzt einen Shard key in dem steht welche Felder sich in diesem Teil der Daten befinden.
Router: Routed die Anfragen auf die spezifischen Shards. Handelt die Kommunikation mit dem Cluster
Config Servers: Speichern Metadaten über den Cluster. Welche Collection sind wo?. Wieviele Shards gibt es?
Wir sehen durch die Documents sind die Daten gut aufteilbar

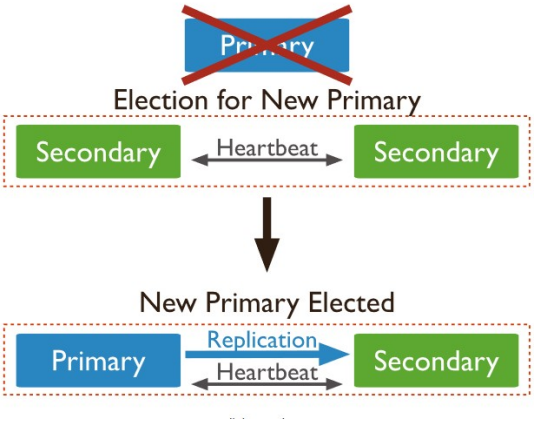
In der Live Demo alles in einem Computer.

Replikation



Gruppe aus mongod instanzen. Mongod ist dabei der Hautpverwaltungsdaemon der MongoDB Datenbank
Primary Node erhält alle schreib Operationen.
Speichert diese in einem Oplog.
Und verteilt dieses oplog weiter zur replikation an die secondaries.
Diese arbeiten das oplog dann asynchron ab.
Wenn ein Replication set erstmals erstellt wird, wird der Datensatz vom Primary geklont.
Die Größe des oplogs ist fest daraus wird abgeleitet ob ein initial sync notwendig ist.
Frage: Was denkt ihr passiert wenn die Primary Node ausfällt?

Replikation



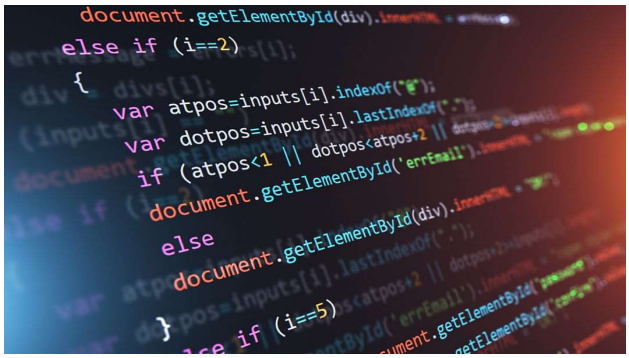
Falls die primary node ausfällt haben die Secondarys einen eingebauten timeout,
Ab wann sie eine neue Wahl zum nächsten primary starten.

MongoDB vs RDBMS

	MongoDB	RDBMS
Skalierbarkeit	✓	✗
Primär- und Fremdschlüssel	✗	✓
Schnelligkeit	✓	✗
Konsistenz	✗	✓
Atomizität	Auf Dokumentebene	✓
Schema	Kann definiert werden	✓

Gesamte Information für ein Dokument muss in diesem Dokument sein.

1. Allgemein
2. MongoDB
3. **Live Demo**
4. Fragen
5. Key Takeaways



1. Allgemein
2. MongoDB
3. Live Demo
4. **Fragen**
5. Key Takeaways

Fragen

Gibt es ein JOIN wie in RDBMS in der MongoDB Abfragesprache?

A: Ja

B: Nein

Nicht ganz:
https://docs.mongodb.com/master/reference/operator/aggregation/lookup/#pipe._S_lookup

Fragen

Was sind die charakterisierenden Eigenschaften eines Dokuments?

- Eindeutiger Key
- Strukturierter Text, Objektähnlich
- Gekapselt

Fragen

Für welche Anforedungen ist MongoDB besonders geeignet?

- Schnellen Release
- Unterschiedliche Daten
- Viele Daten
- Schnelles Sammeln der Daten wichtiger als schnelle Analyse
- Webbasiert

1. Allgemein
2. MongoDB
3. Live Demo
4. Fragen

5. Key Takeaways

Key Takeaways Document Stores

- Geeignet für viele unterschiedliche Daten
- Dokumente als gekapselte objektähnliche Dateneinheit
- Schnelligkeit im Sammeln, im Austausch gegen Schnelligkeit im Analysieren

Quellen

- https://db-engines.com/de/ranking
- https://github.com/ramnes/awesome-mongodb
- https://docs.mongodb.com/manual/
- https://blog.panoply.io/couchdb-vs-mongodb
- https://docs.couchbase.com/home/index.html
- https://docs.couchdb.org/en/stable/
- https://www.mongodb.com/nosql-explained/nosql-vs-sql